Fractional-Order Modelling in Modelica

Alexander Pollok[1] Dirk Zimmer[1] Francesco Casella[2]

[1]Institute of System Dynamics and Control, German Aerospace Center (DLR), Germany,
{alexander.pollok,dirk.zimmer}@dlr.de
[2]Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Italy,
francesco.casella@polimi.it

Abstract

Most dynamic systems with a basis in nature can be described using Differential-Algebraic Equations (DAE), and hence be modelled using the modelling language Modelica. However, the concept of DAEs can still be generalised, when differential operators of non-integer order are considered. These so called fractional order systems have counterparts in naturally occuring systems, for instance in electrochemistry and viscoelasticity. This paper presents an implementation of approximate fractional-order differential operators in Modelica, increasing the scope of systems that can be described in a meaningful way. Properties of fractional-order systems are discussed and some approximation methods are presented. An implementation in Modelica is proposed for the first time. Several testing procedures and their results are displayed. The work is then illustrated by the application of the model to several physically motivated examples. A possible usability-enhancement using the concept of "Calling Blocks as functions" is suggested.

Keywords: Fractional Order Systems, fractional calculus, Integer-Order Approximations

1 Introduction

In Modelica, models are represented as Differential-Algebraic Equations, i.e., equations of the form $F(\dot{x}(t),x(t),t) = 0$. This formulation is adequate for most physical systems that can be described (or at least approximated) with a finite number of states. There are, however, some systems, where a more general but ultimately similar framework is needed: If fractional derivatives occur, the traditional DAE formulation is inadequate.

Fractional calculus, a misnomer[1], is a branch of mathematics that deals with non-integer powers of differentiation operators. The introduction to this concept is much simpler in the Laplace-domain. Normally, the Laplace variable is restricted to integer values. In fractional calculus, this restriction is lifted. Let us imagine the bode diagrams of the derivative operator ($s = s^1$), the unity operator ($1 = s^0$) and the half-derivator ($s^{0.5}$). The amplitude plot of the half-derivator has a slope of 10dB per decade, while the phase angle is constant at 45 degrees. This is illustrated in Figure 1. A detailed discussion of fractional calculus is given by Sabatier et al. (2007).

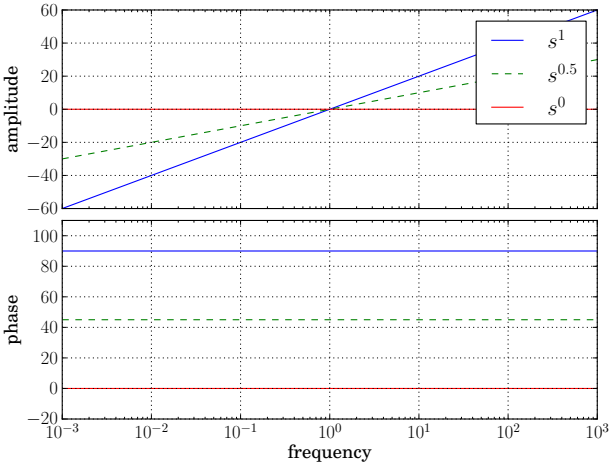

Figure 1. Illustration of fractional derivatives

Going back to time-domain, fractional differential operators can be defined in various ways, with the Caputo definition (Caputo, 1967) being applied in this paper. Contrary to the often preferred Riemann-Liouville definition, the Caputo definition allows for a physically meaningful initialisation of the operator. The Caputo Fractional Derivative of order α is defined as

$$\mathscr{L}^{-1}(s^\alpha) = D^\alpha f(t) := \frac{1}{\Gamma(m-\alpha)} \cdot \int_0^t \frac{f^m \tau}{(t-\tau)^{\alpha+1-m}} d\tau$$

with $f : \mathbb{R} \to \mathbb{R}$ being a continuous, differentiable function, the gamma function Γ, $\alpha \in \mathbb{R}, 0 < \alpha$ and $m \in \mathbb{Z}^+, m = ceil[\alpha]$.

Fractional-order systems occur naturally in various fields, like electrochemistry (Debnath, 2003), viscoelasticity (Koeller, 1984), heat diffusion (Povstenko, 2004) and biology (Magin, 2004). For example, exact solutions

[1]this generalisation of differentiation operators is not restricted to fractions

for local temperature and heat flux at the boundary of a semi-infinite body, using fractional calculus, are given by Kulish and Lage (2000) (for a Modelica Standard Library (MSL)-friendly implementation of this findings see Subsection 3.1). Another application is shaping noise frequency content according to given spectra (Klöckner et al., 2015). The usefulness is not limited to the modelling of PDE's though, fractional order modelling can be used to describe the dynamics of scale-free networks, for instance(Goodwine and Leyden, 2015).

The goal of this paper is to show how fractional-order systems can be modelled using Modelica. The derivation, implementation and testing of suitable approximations in Modelica is illustrated in Section 2. Applications of this implementations are shown in Section 3. At last, the contents of the paper are discussed and possible ramifications to the Modelica languages are adressed in Section 4.

2 Fractional Order Modelling

2.1 Implementation

In Modelica, only the der()-operator has to be generalized to model arbitrary fractional order systems. Unfortunately, defining a new operator fracder(state, order) is not possible based on the Modelica Language Specification 3.3. Therefore, the proposed implementation uses a Single-Input Single-Output block instead.

Fractional order systems have infinite dimensional transfer functions, and therefore have infinite memory (Vinagre et al., 2000). It is, however, possible to find reasonable approximations if the frequency range of interest is bounded (for a detailed error analysis refer to Pan and Das (2012)). Accordingly, as long as the modeller is not interested in extremely stiff systems, these approximations are adequate.

For implementations based on equation-based modelling languages, integer-order continuous approximations are the most useful. For examples, see Carlson and Halijak (1964), Xue et al. (2006) or Oustaloup et al. (2000). Some other methods are described in Vinagre et al. (2000), but not mentioned here, as their implementation in Modelica proved difficult due to a lack of user-level symbolic manipulation capabilities.

For Carlson's method, Oustaloup's method and Xue's method, we found general symbolic expressions for the approximating transfer-functions, and implemented them in Modelica. Preliminary analysis showed that Oustaloup's method was superior regarding flexibility and accuracy. For this reason, in the following only Oustaloup's method and its implementation details are presented.

The integer-order approximation of a fractional operator by Oustaloup's method is given in the Laplace domain by

$$s^{\lambda} \approx G(s) = \omega_h^{\lambda} \cdot \prod_{k=-N}^{N} \frac{s+\omega_k'}{s+\omega_k} \tag{1}$$

$$\omega_k = \omega_b \left(\frac{\omega_h}{\omega_b}\right)^{\frac{k+N+0.5(1+\gamma)}{2N+1}}, \omega_k' = \omega_b \left(\frac{\omega_h}{\omega_b}\right)^{\frac{k+N+0.5(1-\gamma)}{2N+1}} \tag{2}$$

with the fitting range (ω_b, ω_h), the fraction of differentiation λ and the order of approximation N.

An important thing to notice is that λ is not bounded, so it is possible to simulate the second integral using $\lambda = -2$, for example. However, more accurate results can be obtained if $abs(\lambda)$ is kept low and surplus differential operations are simulated directly using the standard Modelica-notation.

We recreated the construction rule for the Oustaloup-Approximator in Modelica using linked first-order elements. The corresponding code can be seen in Listings 1, 2 and 3.

2.2 Testing

To test the validity and accuracy of the derived models, we applied three different testing scenarios: bode diagram, step response, and harmonic displacement. These tests are described in the following.

2.2.1 Bode Diagram

A Bode diagram of the unity operator (1 in the Laplace domain) is a straight line with amplitude 1 and phase angle zero degrees over the complete frequency range. The differentiator (s in the Laplace domain) has an positive slope of 20dB per decade and +90 degrees phase angle. Other operators like s^2 or s^{-1} behave analogous. From this, we require the half-differentiator $s^{0.5}$ to feature an ascending amplitude of 10dB per decade and +45 degrees phase angle.

In Figure 2 and Figure 3, bode plots of the implemented model with approximation orders 2 and 4 and fitting range (0.001Hz,1000Hz) are presented.

It can be seen that slope of the amplitude shows a close fit to the required 10dB per decade. The phase angle shows pronounced ripple effects in the case of the 2$^{\text{nd}}$ order approximation, but no visible ripples in the case of the 4$^{\text{th}}$ order approximation.

The required amplitude values are matched in the complete fitting range. For the phase angle the acceptable range is somewhat smaller.

If the fitting interval is increased to cover a broader range of frequencies (not shown here), noticable ripples in the phase plot appear even for the 4$^{\text{th}}$ order approximation.

Listing 1. Excerpt of Modelica code for a fractional derivative operator

```
block OustaloupOperator
  import Modelica.Blocks.Types.Init;

  parameter Integer order(min=1, max=4) = 4 "order of approximation (1,2,3,4)";
  parameter Real lambda = 0.5 "exponent of operator (-1=integrator, 1=derivative)";
  parameter Real w_lower(max=1) = 0.001 "lower fitting frequency [1/s]";
  parameter Real w_upper(min=1) = 1000 "higher fitting frequency [1/s]";

  parameter Modelica.Blocks.Types.Init initType=Init.InitialState
    "Type of initialization (1: no init, 2: steady state, 3: initial state,
    4: initial output)" annotation(Evaluate=true, Dialog(group= "Initialization"));
  parameter Real x_start[number]=zeros(number) "Initial or guess values of states"
    annotation (Dialog(group="Initialization"));
  parameter Real y_start=0 "Initial value of output"
    annotation(Dialog(enable=initType == Init.InitialOutput, group= "Initialization"));

  final parameter Real wb = w_lower*Modelica.Constants.pi;
  final parameter Real wh = w_upper*Modelica.Constants.pi;
  final parameter Integer number = 1 + order*2;
  final parameter Real K = wh^(lambda);
  final parameter Real wk[number] =
    {FractionalOrder.Approximations.Internal.wk(i,wb,wh,order,lambda)
    for i in -order:order};
  final parameter Real wks[number] =
    {FractionalOrder.Approximations.Internal.wks(i,wb,wh,order,lambda)
    for i in -order:order};

  Real y_internal[number];
  Real x_internal[number];

  Modelica.Blocks.Interfaces.RealInput u
    annotation (Placement(transformation(extent={{-120,-10},{-100,10}})));
  Modelica.Blocks.Interfaces.RealOutput y
    annotation (Placement(transformation(extent={{100,-10},{120,10}})));

equation
  der(x_internal[1]) = -wk[1]*x_internal[1] + (wks[1]-wk[1]) * u*K;
  y_internal[1] = x_internal[1] + u*K;

  for i in 2:number loop
    der(x_internal[i]) = -wk[i]*x_internal[i] + (wks[i]-wk[i]) * y_internal[i-1];
    y_internal[i] = x_internal[i] + y_internal[i-1];
  end for;

  y = y_internal[number];

initial equation
  if initType == Init.SteadyState then
   der(x_internal) = zeros(number);
  elseif initType == Init.InitialState then
    x_internal = x_start;
  elseif initType == Init.InitialOutput then
    y = y_start;
  end if;

end OustaloupOperator;
```

Listing 2. First internal function for the generation of coefficients

```
function wks
    extends Modelica.Icons.Function;
  input Integer k;
  input Real wb;
  input Real wh;
  input Real N;
  input Real lambda;
  output Real wks;
algorithm
  wks:=wb*(wh/wb)^((k+N+(1-lambda)/2)/(2*N+1));
end wks;
```

Listing 3. Second internal function for the generation of coefficients

```
function wk
    extends Modelica.Icons.Function;
  input Integer k;
  input Real wb;
  input Real wh;
  input Real N;
  input Real lambda;
  output Real wk;
algorithm
  wk:=wb*(wh/wb)^((k+N+(1+lambda)/2)/(2*N+1));
end wk;
```

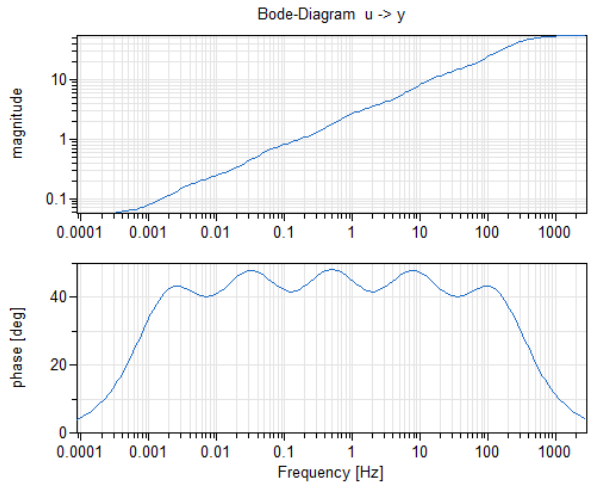

Figure 2. Bode diagram of 2nd order approximation of the half-derivative $s^{0.5}$ with fitting interval (0.001Hz,1000Hz)

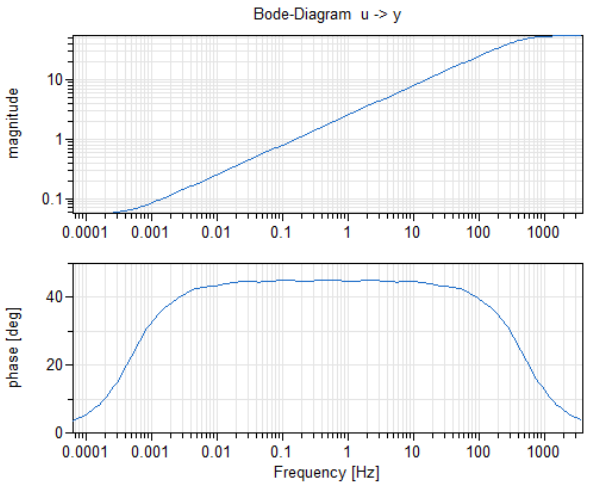

Figure 3. Bode diagram of the 4th order approximation of the half-derivative $s^{0.5}$ with fitting interval (0.001Hz,1000Hz)

2.2.2 Step Response

The step responses of the unity operator $y(t) = u(t)$ and the integrator $\dot{y}(t) = u(t)$ are known to be $y(t) = 1$ and $y(t) = t$ respectively, neglecting the initial conditions. Simultaneously, the unity operator and the integrator are identified in the Laplace Domain by s^0 and s^{-1}. The step responses of the fractional derivatives defined by s^λ with $-1 \leq \lambda \leq 0$ have to constitute the continuous transition between those known step responses (Oldham, 1974).

The implemented model was instantiated 6 times and assigned λ-values in 0.2 intervals between -1 and 0. All models were subjected to a unit step (implemented by setting the input to 1 and the initial states of the models to 0). The results of this test can be seen in Figure 4.

It can be seen that the 6 step-responses form a smooth transition, and the outer ones correspond to the known

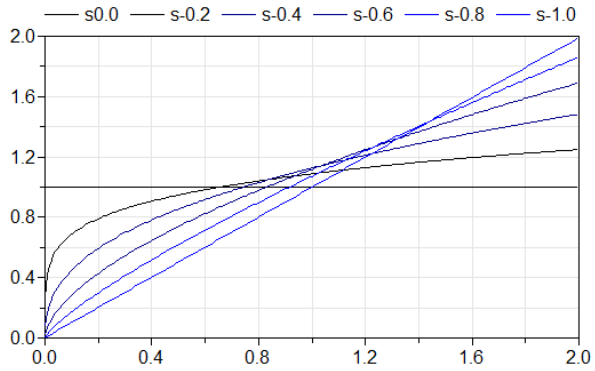

Figure 4. Step responses of $s^{0.0}$, $s^{-0.2}$, $s^{-0.4}$, $s^{-0.6}$, $s^{-0.8}$ and $s^{-1.0}$, approximated with the 3rd order Oustaloup's Method

step-responses mentioned earlier. All curves also match the curves given in Oldham (1974).

2.2.3 Harmonic Displacement

If a harmonic function is derived or integrated, the resulting function is a new harmonic function with a phase offset. For the second test, it is required that the result of fractionally integrated or derived harmonic functions behaves analogous.

The implemented model was instantiated 8 times and assigned λ-values in intervals of size 0.5 between -1 and 2.5. All models were subjected to a cosine input. Initialisation was done in such a way as to avoid unnecessary large initial transients: The models with derivative character would see the onset of the cosine input as a step and correspondingly respond with an impulse. For this reason, their initial states were set to a steady state solution. Models with integrative character were set to zero initial conditions, to set their integration constants to zero as well. The results of this test can be seen in Figure 5.

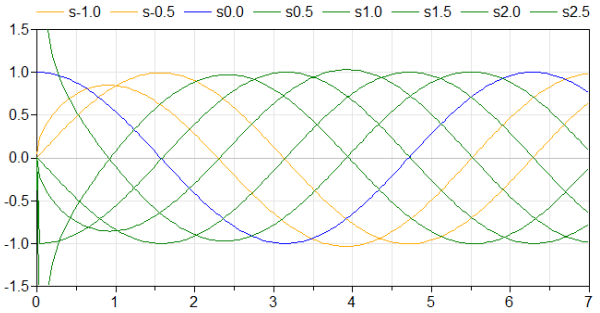

Figure 5. Cosine response of $s^{-1.0}$, $s^{-0.5}$, $s^{0.0}$, $s^{0.5}$ (zero state initialisation) and $s^{1.0}$, $s^{1.5}$, $s^{2.0}$, $s^{2.5}$ (steady state initialisation) approximated with the 3rd order Oustaloup's Method

It can be seen that all model outputs form harmonic functions. Also, the offsets between the functions are uniform. The initialisations of $s^{0.5}$ and $s^{2.5}$ are obviously not optimal, but after a few seconds those deviations vanish.

3 Examples

3.1 Heat Conduction

In Kulish and Lage (2000), relationships between temperature and heat flow rate at arbitrary locations in semi-infinite domains are developed. The temperature at a given time at the boundary is in this way given by

$$T(t) = \frac{\alpha^{1/2}}{2 \cdot A \cdot k} \cdot \frac{\delta^{-1/2} Q(t)}{\delta t^{-1/2}} + T_0 \tag{3}$$

with the thermal conductivity k, the thermal diffusivity α, the Area A, the heat flow rate Q, and the starting temperature T_0.

As can be seen in Listing 4, the corresponding implementation in Modelica is straightforward and compact.

Listing 4. Modelica implementation of a semi-infinite thermal domain

```
  Approximations.OustaloupOperator
    halfInt(order=3, lambda=0.5);
equation
  halfInt.u = heatPort.Q_flow;
  heatPort.T =
   (alpha^(1/2)/(k*A*2)*halfInt.y) + T_0;
```

In Figure 6, the result of a simulation can be seen, where a semi-infinite block was subjected to a periodic rectangular heat flow rate at the boundary. The temperature at the boundary exhibits strong memory-effects, as would be expected from such a system.

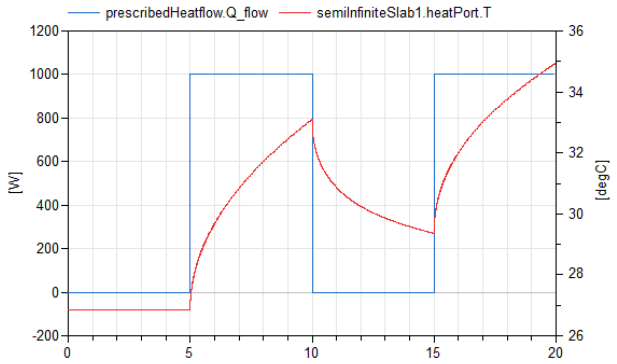

Figure 6. Temperature response of a semi-infinite domain subjected to periodic rectangular heat flow

3.2 Viscoelasticity

The dynamic behaviour of viscous fluids is commonly described with the Navier-Stokes equations. For the dynamic behavior of linear elastic materials, the Lame-Navier equations are used. Both equations have some similarities. As an example, let us take a look at the respective relationships between stress/velocity and

stress/strain for incompressible fluids and linear-elastic solids:

$$\begin{aligned} \boldsymbol{\tau} &= \mu \left(\nabla \mathbf{v} + \nabla \mathbf{v}^T \right) \\ \boldsymbol{\sigma} &= \frac{1}{2} \mathbb{C} \left(\nabla \mathbf{u} + \nabla \mathbf{u}^T \right) \end{aligned} \tag{4}$$

with the stress tensors $\boldsymbol{\tau}$ and $\boldsymbol{\sigma}$, the viscosity μ, the tensor of elasticity $\mathbb{C}$, and the velocity and displacement tensors $\mathbf{v}$ and $\mathbf{u}$. The structure of both equations essentially differs only by one differential operation, as the velocity is the derivative of the displacement w.r.t. time.

Likewise, a relationship between the stress and strain in one-dimensional viscoelastic materials was found in Stiassnie (1979).

$$\sigma = k \cdot \frac{\delta^\alpha \varepsilon}{\delta t^\alpha} \quad (0 \leq \alpha \leq 1) \tag{5}$$

with the stress τ, the material properties k and α, and the strain ε. For α-values of 0 and 1, the behaviour of pure solids and fluids is obtained.

As with the other example, the implementation of this model in Modelica only takes two lines of code (not shown here).

The response of a viscoelastic block ($\alpha = 0.45$) under constant tension can be seen in Figure 7. The result is similar to the results presented by Stiassnie (1979), where the model is validated against real-world measurements.

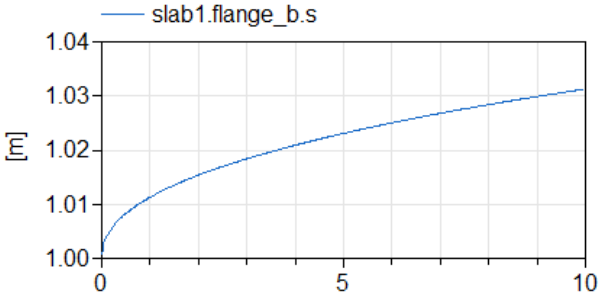

Figure 7. tension-response of a viscoelastic block with a length of 1m under constant tension

4 Discussion

The last sections showed that DAE systems containing time-derivatives of fractional order can be successfully implemented by the given means of the Modelica language. The provided solution offers an approximation that is good enough for at least a large set of technical applications. However, the given examples indicate that the typical application of fractional order derivatives is on the textual modelling level and that the applied formulation is, although practically feasible, still more clumsy than actually necessary. The reason for this is the implementation in block-form. This leads to a declaration that has a dummy character since it is only being used to textually connect its input and output. Because any implementation of a fractional-order operator requires an internal state, an implementation as function is not possible. Yet, it would be very helpful for the modeller if he could call the block like it would be a function. In concrete terms, this means that an anonymous declaration of a block in the equation section is enabled whose inputs and outputs are connected within the declaration statement. To illustrate this mechanism, let us revisit the example of Listing 4:

Listing 5. Modelica implementation of a semi-infinite thermal domain using the "calling blocks as function"-approach

```
block halfInt =
  Approximations.OustaloupOperator
  (order=3,lambda=0.5);
equation
  heatPort.T = (alpha^(1/2)/k)
    *halfInt(u=heatPort.Q_flow).y/A + T_0;
```

Listing 5 presents a reformulation of Listing 3 based on the concept "Calling blocks as function". First, a local declaration of a half-integrater is created from the general Operator-block. Then the expression `halfInt(u=heatPort.Q_flow).y` is represented as an anonymous declaration of the halfInt block. Within the parantheses, the input is connected and the .y states that the expression as a whole represents the output signal of the block. The presented concept "Calling Blocks as Function" is not a new idea. Different syntactical variants are currently in discussion within the Modelica Association based on contributions by Martin Otter, Hans Olsson, Peter Fritzson, Michael Sasena, Martin Sjölund and others. An implementation variant in an experimental equation-based language can be found for instance in Sol (Zimmer, 2010). Should this feature become part of a future Modelica language version, modelling with fractional order time-derivatives will be almost as convenient as with standard time derivatives.

5 Conclusion

By design, the Modelica language is limited to the use of integer-order differential operators. This excludes the modelling of certain physical systems. We present an implementation of Oustaloup's approximation method in Modelica. The resulting model approximates fractional-order differential operators. Parameters for approximation order and frequency fitting range can be used to tailor the model to a specific application. In this way, the mentioned limitation of the Modelica language can be conveniently bypassed, thus increasing the scope of physical systems that can be described in a meaningful manner.

Reproducible research

The results of this paper can be reproduced using the code which is made available on:
github.com/DLR-SR/FractionalOrder

References

Michele Caputo. Linear models of dissipation whose q is almost frequency independent part 2. *Geophysical Journal International*, 13(5):529–539, 1967.

G Carlson and C Halijak. Approximation of fractional capacitors (1/s)ˆ(1/n) by a regular newton process. *Circuit Theory, IEEE Transactions on*, 11(2):210–213, 1964.

Lokenath Debnath. Recent applications of fractional calculus to science and engineering. *International Journal of Mathematics and Mathematical Sciences*, 2003(54):3413–3442, 2003.

Bill Goodwine and Kevin Leyden. Recent results in fractional-order modeling in multi-agent systems and linear friction welding. *IFAC-PapersOnLine*, 48(1):380–381, 2015.

Andreas Klöckner, Andreas Knoblach, and Andreas Heckmann. How to shape noise spectra for continuous system simulation. In *Proceedings of the 11th International Modelica Conference*, 2015.

RC Koeller. Applications of fractional calculus to the theory of viscoelasticity. *Journal of Applied Mechanics*, 51(2):299–307, 1984.

VV Kulish and JL Lage. Fractional-diffusion solutions for transient local temperature and heat flux. *Transactions-American Society of Mechanical Engineers Journal of Heat Transfer*, 122(2):372–375, 2000.

Richard L Magin. Fractional calculus in bioengineering. *Critical Reviews in Biomedical Engineering*, 32(1), 2004.

Keith B Oldham. *The fractional calculus*. Elsevier, 1974.

Alain Oustaloup, Francois Levron, Benoit Mathieu, and Florence M Nanot. Frequency-band complex noninteger differentiator: characterization and synthesis. *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on*, 47(1):25–39, 2000.

Indranil Pan and Saptarshi Das. *Intelligent fractional order systems and control: an introduction*. Springer Publishing Company, Incorporated, 2012.

Yu Z Povstenko. Fractional heat conduction equation and associated thermal stress. *Journal of Thermal Stresses*, 28(1): 83–102, 2004.

J Sabatier, Om P Agrawal, and JA Tenreiro Machado. *Advances in fractional calculus*, volume 4. Springer, 2007.

Michael Stiassnie. On the application of fractional calculus for the formulation of viscoelastic models. *Applied Mathematical Modelling*, 3(4):300–302, 1979.

BM Vinagre, I Podlubny, A Hernandez, and V Feliu. Some approximations of fractional order operators used in control theory and applications. *Fractional calculus and applied analysis*, 3(3):231–248, 2000.

Dingyu Xue, Chunna Zhao, and Yang Quan Chen. A modified approximation method of fractional order system. In *Mechatronics and Automation, Proceedings of the 2006 IEEE International Conference on*, pages 1043–1048. IEEE, 2006.

Dirk Zimmer. *Equation-based modeling of variable-structure systems*. PhD thesis, Swiss Federal Institute of Technology, Zürich, 2010.